

A BIT-LET 46. számát tartja kezében a tisztelt olvasó. S mint-hogy lapocskánk mindig is szerette a rendhagyó dolgokat, úgy gondoltuk, hogy ezt a meglehetősen nem kerek számot használjuk ki arra, hogy olvasóink elé térjünk gondoljainkat, s segítségüket kérjük a megújuláshoz.

Mert mi történt, mi változott meg a hazai, számítástechnikai sajtó olvasó közönség körében az elmúlt négy évben?

(Bizony, ez a 46. szám két szám híján négy év alatt jelent meg! Ennyit tetszettek öregedni, kedves törzsolvások, velünk, szerkesztőkkel együtt!)

Amikor elkezdtek, az egyetlen, akkor szinte még kizárólag a profiknak szóló Számítástechnikán kívül semmiféle homecomputereseknek készülő lap sem volt kis hazánkban. Amikor elkezdtek, alig volt még gép az országban – néhány ezer ABC 80, ZX 81, Spectrum és Commodore 64, HT, valamint néhány száz saját építésű HOMELAB, s pár, alig jegyzett márká. Amikor elkezdtek, bizony minden sor információ kincset ért az olvasók számára.

Az elmúlt években megsokasodtak a lapok, amelyeket az amatőrök megvehetnek, s amelyeket nekik kívánnak eladni. (Örülünk, hogy megsokasodtak.) Az elmúlt években több tízezer gép érkezett az országba, s pár ezret itt is legyártottak. Az elmúlt években kicsit tán zsákutcába is jutott az amatőr számítógépes mozgalom.

Az elmúlt években hál' istennek úgy érezzük a BIT-LET azért megtartotta sajátos karakterét, hangvételét, s ily módon őrzi pozícióját a hazai szakajtóban. Akkor meg mi a gond? – kérdezhetik olvasóink. Nos, úgy érezzük, hogy éppen a mozgalom (ha egyáltalán lehet ezt mozgalomnak nevezni, no de nem találunk jobb szót rá) lendületvesztése, az a tény, hogy egyre élesebben lehet határt húzni a hobbi használatú, s a profi célú



személyi számítógépek közé, megváltoztatja, megváltoztatta a helyzetet. Míg négy évvel ezelőtt hasonló dolgok érdekelték azokat, akik eredeti szakmájukat tekintve ugyan nem voltak számítástechnikusok, de a személyi számítógépek megjelenésekor mérnökként, közgazdászokként ezzel kezdtek el foglalkozni, s azokat, akik diákként, felnőttként tanulták a gép kezelését, programozását, s a legújabb játékok megoldását. Addig ma ez az akkor hasonló érdeklődésű közönség kettévált. Egészen más érdekli a profikká vált programozókat, felhasználókat, s más a diákokat, a hobbi szinten megmaradt érdeklődőket. Úgy gondoljuk, hogy a megváltozott helyzethez alkalmazkodnunk kell. Talán a BIT-LET elmúlt négy éves történetében is érzékelhető volt időnként a változtatás szándéka. De most szükségét érezzük, hogy bizonyos változtatások, új rovatok létrehozása, néhány régi megszüntetése okán szerkesztői jelszavunkhoz híven kikérjük az olvasók véleményét is néhány kérdésben.

Kérjük tehát, hogy mindazok az olvasóink, akik úgy érzik – a jövőben is velünk kívánnak maradni, olvasóink lesznek ezután is, töltsék ki azt a néhány perc alatt kitölthető kis kérdőívecskét, amely a 30. oldalon található. S ha már kitöltötték, adják is postára. Cserébe nem ígérhetünk mást, minthogy a beérkezett cédulákat összeaíjtjuk, s nemcsak közöljük az eredményeket lapunkban, de figyelembe is vesszük munkánkban.

Segítségüket előre is köszöni a szerkesztő:

Angyalosi László

BELÜLRŐL

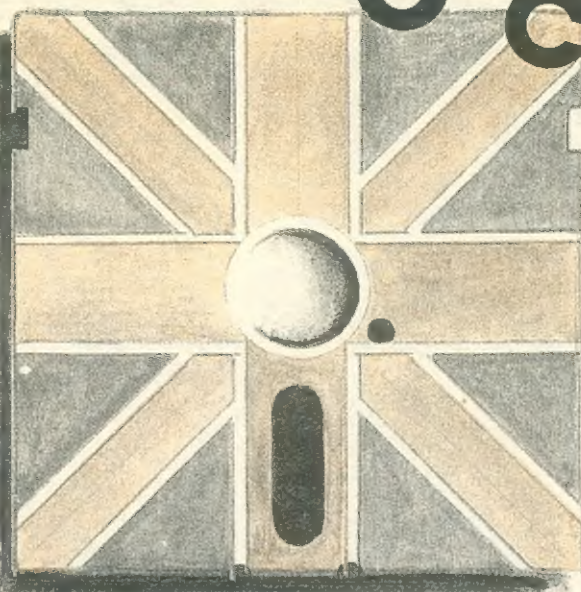
- 18 **Híroldal** – amelyben ezúttal az új IBM családot tekinthetik meg, s mint minden, ami IBM, ez is az érdeklődés középpontjában van világszerte.
- 20 **Exdoctor** – akinek van Commodore drive-ja, annak van rendszerlemez is, s annak megvan ez a program is. Segítünk a kezelésében.
- 22 **Első közből a TV-computeréről** – ez a rész ismét az Editorról szól.
- 24 **Figyelem! Programokat vennénk!** – Hogy milyeneket, azt is megmondjuk.
- 24 **Adatátvitel** – a TVC és a Primo között. Egy speciális felhasználás mikéntje.
- 26 **Teknőcgrafika FORTH-ban** – az első nem BASIC és nem is gépi kódú program a BIT-LET-ben – Spectrumra.
- 28 **Sorvezető** – ezúttal a kombinatórikáról – szakkörök vezetőinek.
- 29 **Posta** – amelyben hosszasan válaszolunk a Plusz/4-es User portjának használatáról szóló kérdésre.
- 30 **Kérdőív** – amelyet reméljük, sokan kitöltenek és visszaküldenek.
- 30 **Hardver ötletek** – Eprom a Spectrumhoz.
- 31 **Könyvmoly** – rágcslálunk egy újabb ismeretterjesztő műre vetette rá magát.
- 32 **Tapasztalatnyerő** – amelyben azért 5000 forintnyi vásárlási utalványt lehet nyerni!

FORDITAS

FORDÍTÁS

Az egész világon foglalkoznak a számítógépes fordítás kutatásával, fejlesztésével, tökéletesítésével. Nálunk két, gépi fordítással kapcsolatos tervezet létezik, de mindkettő kezdeti stádiumban van ahhoz, hogy kész fordító programunk lenne. Budapesten az MTA SZTAKI-ban orosz-magyar, Szegeden angol-magyar fordító rendszeren dolgoznak. A szegediek egy automata szótárkészítő programmal és egy angol alaktani programmal már rendelkeznek.

EXDOCTOR



A Commodor drive-hoz kapott gyári DEMO lemezen szereplő EX-DOS & DISK DOCTOR C 64/VC 1541 programmal, kellő dokumentáció hiányában sok C 64 tulajdonos nem tud mit kezdeni. Pedig ez a program minden amatőr géptulajdonos kincset érő segédprogramja. Az alábbiakban röviden összefoglaljuk azokat a tudnivalókat, amelyek ismeretében mindenki sikeresen használhatja az EXDOCTOR-t.

A program betöltése LOAD "EXDOCTOR", 8, 1 utasítással történik. A sikeres betöltés után az alábbi főmenü látható a képernyőn:

Menu Commands		*	Modul Commands	
F1	Activite Unit 8	*	F1	Flip Page > <
F3	Character Color	*	F3	Last Page
F5	Backgr. Color	*	F5	Next Page
I	Initialize	*	M	Modify
C	Catalog	*	#	Decimal Output
D	Ext. Directory	*	\$	Hex Output
B	BAM	*	H	Hardcopy Unit 4
S	Sectors	*	R	Re-read Data
T	Tracks	*	W	Write Back
X	Buffers	*	↑	Return to Menu
M	Disk CPU Memory	*		
Q	Quit System	*		

MODIFY COMMANDS			
0-9	Decimal Input	Z	Zero Data
#	Hex Input	R	Re-read Data
"	ASCII Input	W	Write back & ^
		A	Return to Module

A PARANCSONK ÉRTELMEZÉSE:

MENU COMMANDS

- F1** Activite Unit 8 – aktiválja a 8-as egységet (a drive-ot)
- F3** Character Color – a karakterek színének kiválasztása
- F5** Backgr. Color – a háttér színének kiválasztása
- I** Initialize – a meghajtóban levő lemez inicializálása
- C** Catalog – directory megjelenítése a képernyőn

– menüparancsok

D Ext. Directory
BAM

S Sectors

T Tracks

X Buffers

M Disk CPU Memory

Q Quit System

- kibővített directory megjelenítése
- foglaltsági térkép kirajzolása a képernyőre
- szektor-(blokk) tartalom megjelenítése
- trekkenként a szektorok első 7 byte-jának kiírása
- lemezbuffer területek megjelenítése
- drive CPU tartalmának megjelenítése
- feldolgozás vége, kilépés a rendszerből

MODULE COMMANDS

F1 Flip Page > <

F3 Last Page

F5 Next Page

M Modify

Decimal Output

\$ Hex Output

H Hardcopy Unit 4

R Re-read Data

W Write Back

↑ Return to Menu

– modulparancsok

- beolvasott lap első, illetve hátsó 126 byte-jának felváltva történő megjelenítése
- lapozás vissza (Láncolási mutató szerint)
- lapozás előre (Láncolási mutató szerint)
- módosító rutin hívása
- memóriatartalom kiírása decimálisan
- memóriatartalom kiírása hexadecimálisan
- képernyőtartalom kinyomtatása
- megjelenített adatok újraolvasatása
- módosított állapotok visszaírása a lemezre
- visszatérés a módosító rutinból a főmenüre

MODIFY COMMANDS

0-9 Decimal Input

\$ Hex Input

" ASCII Input

Z Zero Data

R Re-read Data

W Write back & ↑

↑ Return to Module

– módosító parancsok

- adatbevitel decimálisan
- adatbevitel hexadecimálisan
- bevitel karakteres formában pl.: \$30=48
- adatok nullázása a kijelölt sortól
- megjelenített adatok újraolvasása
- módosított adatok kiírása lemezre és visszatérés az alap üzemmódba
- visszatérés alap üzemmódba

MODULOK RÉSZLETES MAGYARÁZATA:

I (INITIALIZE) funkció:

Időnként előfordul, hogy a diszk esetleges hibája miatt nem tudja elvégezni a szükséges műveleteket. Az INITIALIZE funkció használatával a drive a bekapcsolás utáni állapotot veszi fel.

C (CATALOG) funkció:

Használatával megtekinthető a lemez címkejegyzéke. Csak képernyőfunkció!

D (Ext. Directory) funkció:

Lehetővé teszi a címkejegyzék „kezelését”, különböző információk lekérését a 18-as szektorból.

Lehetséges manipulációk:

Module L U B N T H % R W V Z M

Modify L U B N T H Z R INST DEL

MODULOK ISMERTETÉSE:

L = levédi az összes file-t a törlés ellen

U = törlés elleni védelem feloldása

B = blank-el, SPACE-ekkel feltölti a file nevet 16 karakter hosszúságig

N = blank-elést megszünteti

T = képernyőre írja a programfile-ok betöltési címét, és a programfile-ok végcímét

R = megjelenített adatok újraolvasása

H = (HARDCOPY) nyomtatás

' = megszünteti a programok sorszámozottságát

% = sorszámozza a programokat

W = adott blokk visszaírás a lemezre

V = validálás (nem használt szektorok felszabadítása)

P = megszünteti a törlésre kijelölt file-okat (előtte vissza kell térni a modulba!)

S = P+file-nevek típuson belüli ABC sorrendbe rendezése

M = módosító rutin hívása

Lehetséges módosítások:

FIGYELEM! A módosítás mindig a kiválasztott file-ra vonatkozik.

L, U, B, N, T, R, mint a modul üzemmódban.

DEL = kiválasztott file törlése a címkejegyzékből

INST = kiválasztott file elé beszúrás (üres hely előállítás a címkejegyzékben)

Z = törli az adott file-t és az összes utána következőt

↑ = visszatérés a modulba (Címkejegyzék kiírása csak a modulba)

B (BAM), LEMEZFOGLALTSÁGI TÉRKÉPET RAJZOL

" = szabad blokk

***** = foglalt blokk

s = törölt blokk

F3 = lapozás előre

F5 = lapozás hátra

Módosítási lehetőségek:

E = teljes térképet ad, azaz kijelzi a törölt blokkokat is

A = ALLOCATE, szabad szektorok lefoglalása

F = FREE, szektorok felszabadítása a BAM-ban. Kiválasztás CRSR billentyűvel

Az F3, F5, M, R, W és a felfelé nyíl hatása analóg a már leírtakkal. A H parancs a képernyő tetejétől nyomtatja a BAM térképet a BAM végéig.

S (SECTORS) FUNKCIÓ (SZEKTORTARTALMAK)

Egy adott blokk képernyőre írása (két részletben) és az adott blokk módosítása.

A modul hívásakor elsőként a lemez 18/0 szektorának az első felét írja ki. Minden szektor csak két részletben fér el a képernyőn.

F3 = lapozás előre

F5 = lapozás hátra

Módosítási lehetőségek:

T = Trace Links bekapcsolása lehetővé teszi az 'F' és 'L' funkció használatát

F = First Block, trace után: lánc első elemének megjelenítése

L = Last Block, trace után: lánc utolsó elemének megjelenítése

C = Copy Block – a kiválasztott blokk átmásolása tetszőleges helyre

N = Next non-zero Block, a következő nem üres szektor megjelenítése

M = módosító rutin hívása

Módosítás byte-onként, vagy Z (teljes szektor nullázása).

A byte-ok kiválasztása a CRSR billentyűvel történik.

W = módosított állapot visszaírása a lemezre

Monitorszerű alkalmazás „SECTORS” módban:

Modify módot kell választani, majd " (idézőjel) beütését követően, a „Value”: után kell beírni az új, bevendő karaktert. Az „R” funkció itt is működik!

T (TRACKS) FUNKCIÓ SÁVTARTALMAK):

T (TRACKS) funkció (sávtartalmak):

A kiválasztott sáv szektorainak első 7 byte-ja jelenik meg.

F3 = lapozás előre

F5 = lapozás hátra

Módosítási lehetőségek:

C = teljes sávtartalom másolása tetszőleges helyre

F = teljes sáv újraformázása adatvesztés nélkül

W = az adott track újraformattálása

Megjegyzés: Teljes lemez újraformázás NEW paranccsal.

X (BUFFERS) FUNKCIÓ (LEMEZBUFFEREK):

F1 = flip-flop

F3 = lapozás előre

F5 = lapozás hátra

Módosítás byte-onként, vagy Z (adatok nullázása)

M (DISK CPU MEMORY) funkció (drive memóriatartalom – ROM)

Hardcopy lehetséges.

közreadta: Honti Tamás

PROGRAM CSERE-BERE

C 64-es géphez eredetileg mellékelte új használati utasítást cserélnék német nyelvűről angol nyelvűre. Várom hasonló nyelvi problémával küszködő sorstársaim leveleit. Németül tudók, akiknek angol nyelvű gépkönyve van írjanak.

Holbok Ferenc, 9100 Tét, Szabadság u. 24.

C 16, C 116, Plus 4 számítógépre orosz és angol nyelvű oktatóprogramok eladók. Játékprogramok cseréje.

Kálmán Albert, 3330 Eger, Rákóczi út 31. III. 11.

Tel. üzenet: 143-031, 330-345 (Bp.)

Keresek Sinclair, Sinclair User, és egyéb számítástechnikai újságokat ZX Spectrum cikkekkel és programokkal.

Gedő Tamás, 1165 Bp., Veres Péter út 121. Tel.: 838-037

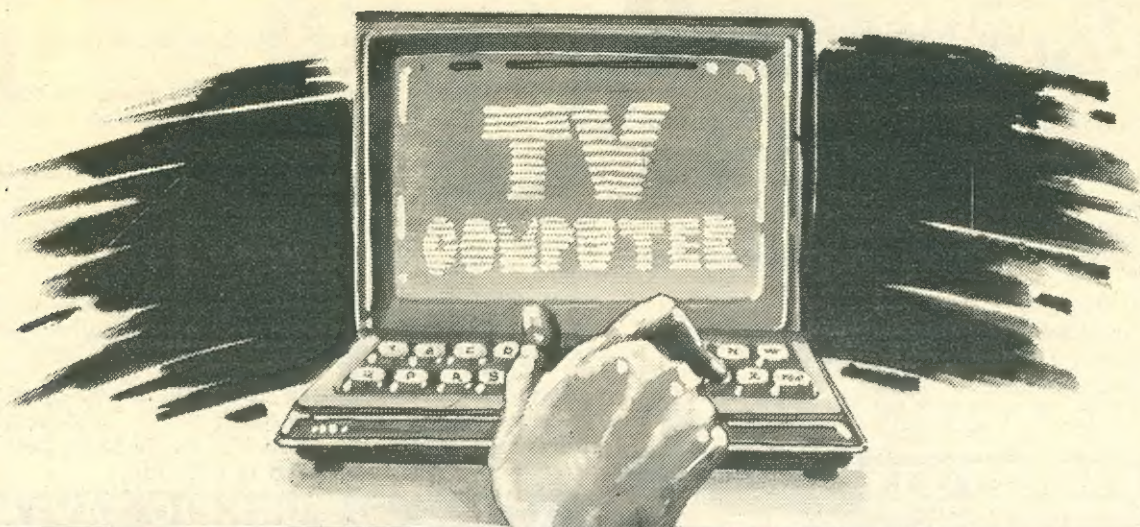
C 64-re cserélnék programot levélben vagy személyesen. Csak kazettán.

Szabó Zoltán, 1211 Bp., Táncsics M. út 65. 1/4.

C 16-os programokat cserélek!

Székely Tamás, 1054 Bp., Rosenberg hp. u. 21. I / 2.

TV-computer játékok és felhasználói programokat cserélnék. A programlistát rövid tartalommal az alábbi címre kérem: Tömöri Zoltán, 1148 Bp., Fogarasi út 28-54.



ELSŐ KÉZBŐL

A TV COMPUTER RŐL

AZ EDITORRÓL

AZ EDITOR RUTINJAI

ED INT hívási kód: 32 vagy 160 (20h vagy A0h)

működés: Az editor interrupt rutin villogtatja a kurzort, amíg az editor karakterek begépelésére vár. A felhasználó számára nem ajánlott a hívás!

ED CHIN hívási kód: 161 (0A1h)

output: C-karakterkód
A-hibakód

működés: Egy karakter beolvasása. Híváskor az editor belép a karakterbeolvasó-szerkesztő programhurokba: a leütött karaktereket beolvassa a billentyűzetről és kiírja a képernyőre, a szerkesztési funkciókat pedig végrehajtja. Háromféleképpen lehet a szerkesztőhurokból kilépni:

ESC-vel: C=27 (1Bh, ESC)

CTRL+ESC-vel: C=27 (1Bh, ESC)
A=245 (0F5h, STOP)

RETURN-nel: C=a bekezdés első karaktere

Ez a harmadik a normál kilépési mód. Ilyenkor a soronkövetkező ED-CHIN hívások a bekezdés további karaktereit kapják vissza C-ben. Ha a karakterek elfogytak, akkor C-ben 13-mal (0Dh, RETURN) tér vissza a rutin, majd egy újabb hívás hatására ismét belép a szerkesztő hurokba. A RETURN kódjának visszasadásakor a kurzor a következő bekezdés, vagy üres sor elejére áll. A képernyő alján scrollozás történik.

ED CHOUT hívási kód: 33 (21h)

input: C=karakterkód

működés: Egy karakter kiírása. A 32-323 kódú (20h-0DFh) karaktereket kiírja a kurzor pozíciójába a képernyőre és a kurzort eggyel jobbra lépteti. A sor végéről a következő sor elejére kerül a kurzor. A képernyőn levő karaktereket felülírja mindaddig, amíg a kurzor egy bekezdés belsejében van. Ha a kiírás eléri a bekezdés utolsó sorának utolsó pozícióját, akkor egy üres sort hozzáfűz a bekezdéshez, és annak az elejére lép. Ha a soronkövetkező sor nem üres, akkor innen kezdve a sorokat eggyel lejjebb lépteti, és beszúrja az üres sort. A kép-

ernyő utolsó sorában természetesen scrollozás fog történni.

A szerkesztési funkcióval rendelkező kódok hatása olyan lesz, mintha azokat a billentyűzetről vittük volna be input során. Három kódnak van az inputtól eltérő jelentése:

10 (0Ah, LINE FEED): A kurzort a bekezdést követő első sorba viszi, az oszloppozíciót nem változtatja meg. A képernyő alján scrollozást hajt végre.

13 (0Dh, RETURN): A kurzort a képernyő sor első pozíciójába viszi.

27 (1Bh, ESC): Itt hatástalan.

Az aktuális tintaszín és háttérszín nemcsak a kiírásnál, hanem a szerkesztési műveleteknél is érvényesül: a beszúrt üres sor pl. háttérszínű lesz.

ED BKIN hívási kód: 162 (0A2h)

input: BC=a beolvasandó karakterek száma
DE=pufferterület kezdőcíme

működés: Karaktercsoport beolvasása. Ez a funkció az ED-CHIN ismételt hívásával működik. Ha az elsőnek beolvasott bekezdés a záró RETURN-nel együtt kevesebb karaktert tartalmaz, mint a BC-ben adott érték, akkor újból belép a szerkesztő ciklusba, és további karakterekre vár. A bekezdést záró RETURN kódját is tárolja. A beolvasott karakterek a DE-ben megadott címtől kezdve találhatók meg a memóriában. Ha a szerkesztés során ESC-t nyomunk, akkor annak kódja, 27 (1Bh) egyből bekerül a pufferterületre. Ha az utolsó bekezdés több karaktert tartalmaz, mint amennyit be akarunk olvasni, akkor a következő ED-BKIN hívás először a félbehagyott bekezdést fogja beolvasni, és csak ennek végeztével lép a szerkesztési ciklusba.

ED BKOUT hívási kód: 34 (22h)

input: BC=kiírandó karakterek száma
DE=pufferterület kezdőcíme

működés: Karaktercsoport kiírása. Ez a funkció az ED-CHOUT ismételt hívásával kiírja a BC-ben megadott számú karaktert a DE-ben megadott memóriacímtől kezdve.

CPOS hívási kód: 35 vagy 163 (23h vagy 0A3h)

input: B=oszloppozíció (0,1-16,1-32,1-64)
C=képernyősor (0,1-24)

output: A=hibakód

működés: Pozícionálja a kurzort a B és C regiszterek értéke alapján. Ha a regiszterbe írt érték zérus, akkor a megfelelő pozíció változatlan marad. A maximális oszloppozíció a grafikus felbontástól függően 16, 32 vagy 64.

CFIX hívási kód: 36 vagy 164 (24h vagy 0A4h)

működés: Kurzorpozíció megjegyzése. Közvetlenül az ED-CHIN vagy ED-BKIN hívások előtt kell meghívni. Hatására az editor megjegyzi az aktuális kurzorpozíciót, és az input során ezt tekinti a paragrafus első beolvasható karakterének. Ezt használja az INPUT PROMPT utasítás is. Csak akkor érvényesül a hatása, ha az input szerkesztő ciklusában nem kerül a kurzor a paragrafusban a megjegyzett pozíció elé, emellett a képernyőn nem mozdult el a megjegyzett pozíció (pl. sortörés vagy sorbeszúrás esetén is elmozdulhat az egész bekezdéssel együtt). Természetesen az input lezárásakor (RETURN) a kurzornak a megjelölt bekezdésben kell lennie. Csupán az első szerkesztési ciklusban veszi az input figyelembe, tehát ED-BKIN esetén a második paragrafus bekérésekor már nem. A szerkesztőciklusban leütött ESC is megszünteti a hatását.

EGY APRÓ FOGÁS:**PROGRAMSOROK****EGYBESZERKESZTÉSE**

Az editor ismertetésénél kiemeltük, hogy két bekezdést, például két kilistázott programsort, nem tudunk normál módon összefűzni. Normál módon nem, de egy trükkel igen! Ehhez a következőket kell tudni:

Az editor a munkaterületén nyilvántartja minden képernyősorról, hogy az hány karaktert tartalmaz. Erre 24 byte-ot használ. Sőt itt jelzi azt is, hogy a sornak van-e folytatása vagy sem. Ha a hosszbyte legfelső bitje 1-es, akkor a bekezdés a következő sorban folytatódik. Ezek alapján egyszerűen össze tudunk fűzni két bekezdést, csak a fenti táblázat kezdőcímét kell ismerni:

LINE TAB 24 byte, címe 3664=0E50h

Négyszínű módban a sorhossz 32 karakter, a táblázatba írandó érték: 32+128 (tele sor+folytatás):

POKE 3663+SOR, 160

A SOR változó helyére a kívánt sor számát (1-24) kell írni. Például BASIC programunk tartalmazza a következő két sort:

100 Y=960-40*Y:X=32*X-32

110 PLOT ,X,Y;X+28,Y;X+28,Y+36;X,Y+36;X,Y

Ebből akarunk egyetlen 100-as sort készíteni. Töröljük le a képernyőt és listázzuk ki ezt a két sort:

CLS:LIST 100,110

Most a képernyő első sorában van a 100-as sor. Az összefűzés

POKE 3663+1,160

Ezután a kurzort az első sor végére visszük, odaírunk egy kettőspontot (:), majd addig ejtjük ki innen a karaktereket (SHIFT+DEL), amíg a PLOT utasítás P-je a kurzor pozíciójába kerül. A sort RETURN-nel lezárjuk, és kitöröljük a feleslegessé vált 110-es sort:

DELETE 110

Végül javaslok mindenkinek kipróbálásra, hogy mi történik akkor, ha az eredeti sorhosszhoz adjuk hozzá a folytatást jelentő 128-at:

POKE 3663+SOR,PEEK(3663+SOR)

Cseh Tibor

MEGHÍVÓ

A csákvári Kossuth Művelődési Házban működő mikroklub és a TIT Fejér megyei szervezete harmadik alkalommal rendez találkozót a számítógépbarátok részére. Rendezvényünkre az évtizedes múltú tekintő csákvári napok keretében kerül sor. Hívjuk és várjuk mindazokat, akiket a mikroszámítógépek érdekelnek.

Szeretnénk ismét találkozni a **COMMODORE C 16, Plus/4, C64, a SINCLAIR SPECTRUM és VIDEOTON TV COMPUTER** gépekkel dolgozó társainkkal, és mindazokkal, akik még csak ismerkednek a számítógéppel.

A találkozón **profi programozók** is részt vesznek, és készséggel állnak minden érdeklődő rendelkezésére.

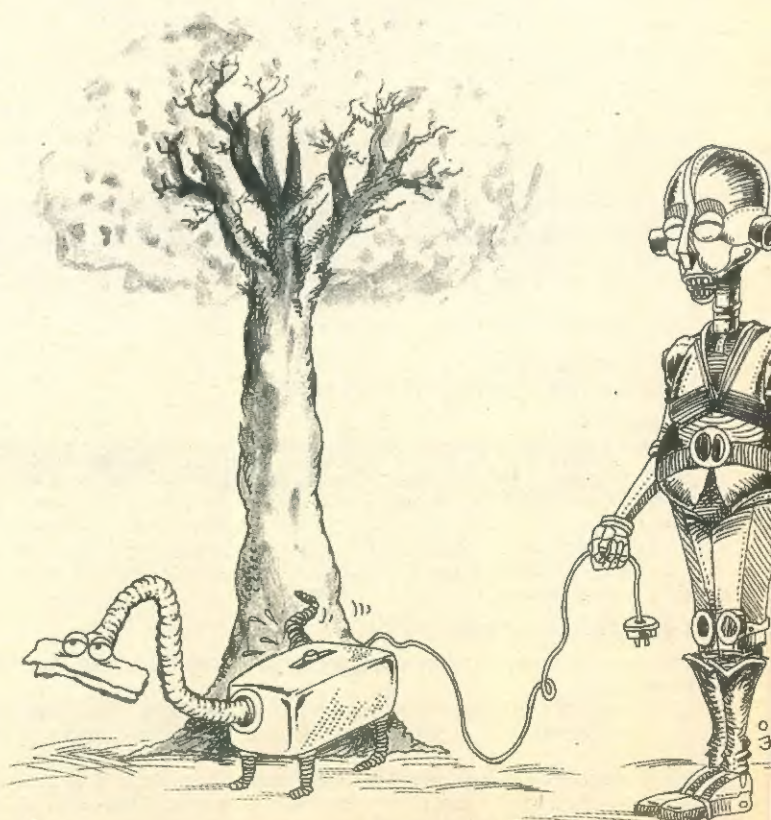
Természetesen rengeteg **játékprogramot** is kipróbálhatnak a résztvevők.

A program során lehetőséget biztosítunk a **szoftverek csereberéjére** is.

Rendezvényünk meglepetése: a VIDEOTON IBM AT kompatibilis gépeinek a bemutatkozása!

A találkozó színhelye a csákvári SPORT-KOMBINÁT. A program időpontja: **1987. augusztus 22.** A bemutató és a cserebere reggel 10 órától délután 5 óráig tart.

A Rendezők



figyelem

PROGRAMOKAT
VENNÉNK!

Akik olvassák és fejtegetik gép- és egyéb nyerő pályázatainkat, tudják, hogy volt egy „Van egy ötlete” című pályázatunk, amelyre programötleteket kértünk. Kaptunk is egy sor értelmes ötletet. Ezek egy részét már közöltük a múlt hónapban, más részét most tesszük közzé. Kérjük, hogy akinek van kedve és ideje ezek valamelyikét megírni, s lapunkban közzétenni vagy egyszerűen küldje be a programot, vagy ha a vállalkozónak előzetes megbeszélőnivalója lenne a dologgal kapcsolatban, hívja föl a BIT-LET szerkesztőjét a 408-603-as telefonon. Reméljük, hogy az ötletek legérdekesebbjeit viszontláthatják majd lapunkban.

SZUHÁR GABRIELLA ÖTLETEIBŐL:

IQ-TEST

Talán szórakoztató lehet. A Raven-féle generálintelligencia-teszt alapján készíteném el, a feladatsor végén közölné az intelligenciahányadost. 48 Kbyte-tól.

GYERMEK-KRESZ ÚTVESZTŐ

Az útvessztő játék változata, a helyes útvonalon minden jól végrehajtott manőver (a tábla utasításai szerint szabályos) után pont jár, a hibáért több mínuszpont levonás kerül. A pályát lehet gyalogosan és kerékpárral teljesíteni.

RÓKA SÁNDOR ÖTLETEIBŐL:

AEROBIC/TV-TORNA

Tornagyakorlatokat bemutató, generáló program, helyettesíthetné a TV-torna adását, s nem lenne műsoridőhöz kötve.

KERESZTREJTVÉNY-KÉSZÍTŐ

Ez nehéz feladat, csak a rejtveny-készítő programot kell elkészíteni, a szóképzetet tartalmazó adatállományt a felhasználó állítja össze magának. A program feladata keresztrejtvény összeállítása az adatállományban levő szavakból, a rejtveny ábráját is a gép generálja.

JUHÁSZ IMRE ÖTLETEIBŐL:

A GYERMEKEK TÉRLÁTÁSÁT SEGÍTŐ, fejlesztő térbeli rajzok készítését segítő program, amely billentyűzetről, vagy joystickról egyaránt működtethető.

KRESZ SZIMULÁCIÓS JÁTÉK, melyben a gyerekek mint gyalogosok, vagy kerékpárosok szerepelnek és konkrét helyzetekben a közlekedési szabályoknak megfelelően dönthetnek. Ez sokat segítené a szabályos közlekedés tanításában a gyerekek között.

AZ EMBERI TEST működését, felépítését bemutató program, mely az általános bemutatás mellett konkrét rákérdezéssel egy-egy szerv működését is bemutatja rajzos formában és folyamatos mozgást szimulálva (pl. szívdobogás).

A PETÁK TAMÁSTÓL ÉRKEZETTEKBŐL:

A KÖZISMERT TORPEDÓ JÁTÉK. A gép is, az ember is rejtse el hajókat, melyekre löni lehet. Ugyanez család változatban: a gép csak akkor helyezze el a hajóit, ha már muszáj.

TÁBLÁS JÁTÉKOK:

- DÁMA
- FRANCIA SAKK
- NE NEVESS KORÁN (ÉS TÁRSAI)

És végül

KISS ÁRPÁD PÁLYAZATÁBÓL:

SPRITE MOZGATÁSA gépi kódú rutin segítségével: (C64) mozgása előre meghatározható legyen (pl. az X, Y ponttól a Z, F pontig és vissza), sebessége beállítható. Így pl. csak a saját emberkém mozgását kellene megoldanom (nagyobb sebességgel mozogni 1 sprite BASIC-ből is mintha 8 sprite-ot egyszerre szeretnék mozgatni), és az ütközéseket kellene leolvasnom.

LEMEZKATALÓGUS:

Beolvasná a lemez tartalomjegyzékét (a lemez neve, ID a programok nevei, blokkok)

Funkciói:

- a) Megadom a program nevét megmondom melyik lemezem, melyik oldalán található. A programokat ABC sorrendben printerre lehessen küldeni, írja mellé a lemez nevét, oldalát.
- b) A programok legtakarékosabb elhelyezése 1 lemezoldal segítségével. Miután beolvassa a tartalomjegyzékét, kérdezze meg mely programok tartoznak össze, és ezt tiszteletben tartva adja meg a legtakarékosabb elhelyezést, printerre is küldhető legyen.

Oktatási intézményeinkben több helyen megtalálhatjuk a TV-Computer és a PRIMO típusú számítógépeket. Az alábbi program segítségével a TVC-n írt adatokat tartalmazó magnetofonkassztát a PRIMO képes beolvasni további műveletekhez.

A TVC-n az alábbi programrészlettel tudunk adatokat a kazettára írni:

```
100 OPEN 5: OUTPUT "Név"
105 PRINT 5: ... adatok ...
110 CLOSE 5: OUTPUT
```

A TVC az adatfile-t ekkor a szalagon szekvencionálisan úgy rögzíti, hogy először egy fejlécet, majd 256 byte-os adatblokkokat ír fel a szalagra. A fejlécet és az adatblokkokat egy-egy szinkronizáló szakasz vezeti be. Ezt követi az adatfolyamat bitenként a blokk végéig. A fejléc és az adatblokkok első byte-ja 0, a második 106 értékű. A fejléc 9. byte-ja tartalmazza a file nevének hosszát, amit a név karakterei követnek. Az adatblokkoknál a 7. byte tartalmazza a blokk sorszámát (az első 1), a 8. pedig a blokkban levő byte-ok számát. Az adatblokkok utolsó byte-ja jelzi, hogy vége van-e már az adatsorozatnak. Ha ez 0, akkor még folytatódik, ha 255, akkor vége.

A PROGRAM BEÍRÁSA

Gépeljük be a PRIMO-ba az 1. számú programot! A kihagyott részekre értelemszerűen a 2. számú lista adatai kerüljenek. A program futtatásakor először beolvassa, majd kiírja a szalagon található adatfile nevét. Ezt követően beolvassa az egyes adatblokkokat és sorszámaikat nagybetűkké alakítva A-tól kezdve megjeleníti. A nagybetűk folyamatos sorából láthatjuk, hogy a beolvasás hibátlan volt-e. Az utolsó blokk után a program visszatér BASIC-be és a 20480-as címtől kezdve az A változó értékéig terjedően megtalálhatjuk a TVC-n létrehozott adatokat. A program átalakítja a TVC-n használt ékezetes karaktereket a PRIMO kódjainak megfelelően, így az előbbin többletként meglévő hosszú ékezetes nagybetűk rövidkévé változnak.

ÉS MÉG...

Néhány szót még a TVC által kiírt numerikus adatokról és az adatkivitel szerkezetéről. A TVC pl. a -101 számot a 45/49/48/49/32 karaktersorozat formájában rögzíti a szalagon. Pozitív számok esetén az előjel helyén egy szóköz (ASCII 32) van. Ha az adatkivitelnél az értékek közé "-"t raktunk, akkor a szalagon itt egy TAB (ASCII 9) kód lesz majd. A TVC minden RETURN (ASCII 13) után egy LF (ASCII 10) karaktert is elhelyez.

Markó György

a tv computer és a Primo között

2. SZÁMÚ LISTA

FEJLÉC BEOLVASÁSA

20000	33	0	80	LD HL,20480	
	205	142	78	CALL SZINKRON	
	22	7		LD D,7	
	205	192	78	CALL BYTE	
	21			DEC D	
	32	250		JR NZ F-1	D = név hossza
	87			LD D,A	
	62	7		LD A,7	
	130			ADD A,D	A = névhossz+7
	245			PUSH AF	
	205	192	78	CALL BYTE	
	119			LD (HL),A	(HL) = név betűi
	35			INC HL	
	21			DEC D	
	32	248		JR NZ F-2	
	54	13		LD (HL),13	végén RETURN
	209			POP DE	D = névhossz+7
	33	250	78	LD HL,20474	kírási kezdőcíme
	126			LD A, (HL)	A = kírándó
	205	232	78	CALL ÉKEZETES?	
	205	199	53	CALL DSPHND	
	35			INC HL	
	21			DEC D	
	32	245		JR NZ F-3	

ADATBLOKK BEOLVASÁSA

20044	33	0	80	LD HL,20480	HL = kezdőcím
	205	142	78	CALL SZINKRON	
	22	5		LD D,5	
	205	192	78	CALL BYTE	
	21			DEC D	
	32	250		JR NZ A-2	
	245			PUSH AF	A = blokk soroz.
	205	192	78	CALL BYTE	
	87			LD D,A	D = blokk hossza
	205	192	78	CALL BYTE	
	119			LD (HL),A	(HL) = adatok
	35			INC HL	
	21			DEC D	
	32	248		JR NZ A-3	
	205	192	78	CALL BYTE	
	71			LD B,A	B = záró byte
	241			POP AF	
	198	64		ADD A,64	A = sorszám+64
	205	199	53	CALL DSPHND	
	120			LD A,B	A = záró byte
	254	255		CP 255	
	32	217		JR NZ A-1	

ÁTALAKÍTÁS

20086	43			DEC HL	HL = utolsó cím
	183			OR A	Carry-be 0
	17	0	80	LD DE,20480	
	237	82		SBC HL,DE	HL = adatok száma
	26			LD A,(DE)	
	205	232	78	CALL ÉKEZETES?	
	18			LD (DE),A	
	19			INC DE	
	43			DEC HL	
	124			LD A,H	
	181			OR L	
	32	245		JR NZ AT-1	
	235			EX DE,HL	HL = utolsó cím
	201			RET	

SZINKRON

20110	30	4		LD E,4	
	119		S-1	LD (HL),A	(HL) = jelhossz
	8	0		LD B,0	
	219	31	S-2	IN A, (magnó)	
	4			INC B	B = üres hossza
	163			AND E	
	32	250		JR NZ S-2	
	14	0		LD C,0	
	219	31	S-3	IN A, (magnó)	
	12			INC C	C = jelek hossza
	163			AND E	
	40	250		JR Z S-3	
	120			LD A,B	
	129			ADD A,C	A = B+C
	254	52		CP 52	kiseb 52-nél
	56	233		JR C S-1	
	254	61		CP 61	nagyobb 60-nál
	48	229		JR NC S-1	A = előző érték
	126			LD A, (HL)	
	254	28		CP 28	kiseb 28-nél
	58	224		JR C S-1	
	254	41		CP 41	nagyobb 40-nél
	48	220		JR NC S-1	
	205	192	78	CALL BYTE	
	205	192	78	CALL BYTE	
	254	106		CP 106	
	32	210		JR NZ S-1	mégsem adatfile
	201			RET	

BYTE BEOLVASÁSA

20160	8	8		LD B,8
	205	200	78	CALL BIT
	16	251		DJNZ BY-1
	201			RET

BIT BEOLVASÁSA

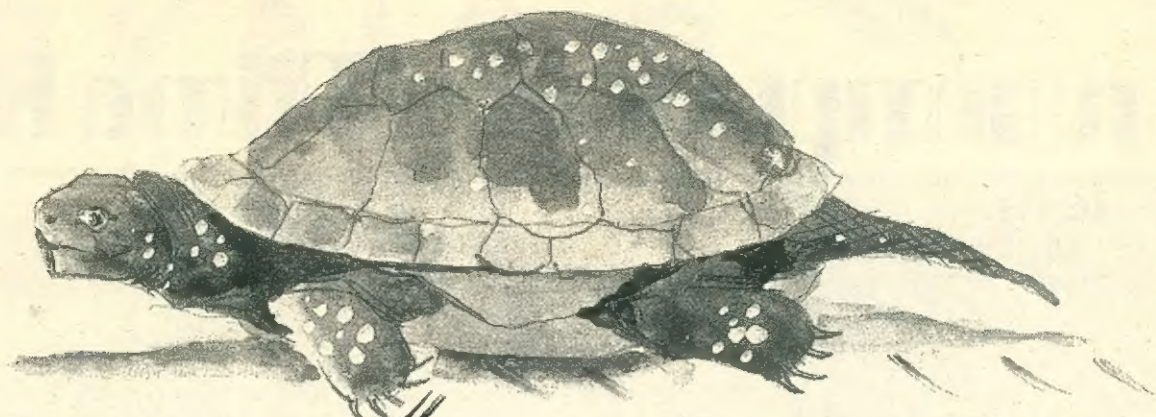
	197			PUSH BC
	245			PUSH AF
	8	0		LD B,0
	219	31	BI-1	IN A, (magnó)
	4			INC B
	163			AND E
	32	250		JR NZ BI-1
	14	0		LD C,0
	219	31	BI-2	IN A, (magnó)
	12			INC C
	163			AND E
	40	250		JR Z BI-2
	120			LD A,B
	129			ADD A,C
	254	34		CP 34
	203	17		RL C
	241			POP AF
	203	25		RR C
	31			RRA
	193			POP BC
	201			RET

ÉKEZETES BETŰK?

20200	254	128		CP 128	
	216			RET C	nem ékezetes
	32	2		JR NZ É-1	nam Á
	62	93		LD A,93	
	254	129	É-1	CP 129	
	32	2		JR NZ É-2	nem É
	62	64		LD A,64	
	254	130	É-2	CP 130	
	32	2		JR NZ É-3	nem I
	62	73		LD A,73	
	254	131	É-3	CP 131	
	32	2		JR NZ É-4	nem Ó
	62	79		LD A,79	
	254	132	É-4	CP 132	
	32	2		JR NZ É-5	nem Ő
	62	92		LD A,92	
	254	133	É-5	CP 133	
	32	2		JR NZ É-6	nem Ő
	62	92		LD A,92	
	254	134	É-6	CP 134	
	32	2		JR NZ É-7	nem Ü
	62	85		LD A,85	
	254	135	É-7	CP 135	
	32	2		JR NZ É-8	nem Ü
	62	94		LD A,94	
	254	136	É-8	CP 136	
	32	2		JR NZ É-9	nem Ú
	62	94		LD A,94	
20255	254	144	É-9	CP 144	
	32	2		JR NZ É-10	nem á
	62	125		LD A,125	
	254	145	É-10	CP 145	
	32	2		JR NZ É-11	nem é
	62	96		LD A,96	
	254	146	É-11	CP 146	
	32	2		JR NZ É-12	nem í
	62	30		LD A,30	
	254	147	É-12	CP 147	
	32	2		JR NZ É-13	nem ó
	62	91		LD A,91	
	254	148	É-13	CP 148	
	32	2		JR NZ É-14	nem ő
	62	124		LD A,124	
	254	149	É-14	CP 149	
	32	2		JR NZ É-15	nem ő
	62	123		LD A,123	
	254	150	É-15	CP 150	
	32	2		JR NZ É-16	nem ú
	62	95		LD A,95	
	254	151	É-16	CP 151	
	32	2		JR NZ É-17	nem ü
	62	126		LD A,126	
	254	152	É-17	CP 152	
	192			RET NZ	nem ű
	62	127		LD A,127	
	201			RET	

1. SZÁMÚ PROGRAM

10 POKE 20000,33,...
 15 POKE 20044,33,...
 20 POKE 20086,43,...
 25 POKE 20110,30,...
 30 POKE 20160,6,...
 35 POKE 20200,254,...
 40 POKE 20255,254,...
 45 POKE 20474,70,105,108,101,58,32
 50 A=CALL(20000)



teknőc-grafika forth-ban

ZX SPECTRUM

Azt gondoltuk, itt az ideje, hogy a **BIT-LET** is kilépjen a **BASIC** bűvköréből. Íme az első nem **BASIC** és nem is gépi kódú program. **FORTH** nyelven íródott, mégpedig **Spectrumra**. Minthogy az ő **Spectrumosok** közül nagyon sokan rendelkeznek már a géphez íródott **FORTH**-szal, úgy gondoltuk bátran közreadhatunk egy ebben a nyelvben írott programot.

A program a teknőcgrafikát ismeri. Konkrétan a **ZX Spectrum Spec Forth** interpreteréhez íródott, de egész kis munkával máshoz is átírható.

A **FORTH** nyelv tulajdonképpen egy olyan szótár, amelyben az egyes szavak bizonyos utasításokat jelentenek. Új szavakat a régiek segítségével lehet definiálni, ezek segítségével még újabbakat stb. Egy **FORTH** program nem más, mint bizonyos számú definíció megadása – épp ezért bármely **FORTH** programhoz lehet mindig hozzáírni még. A **FORTH** utasítások egy ún. paraméter stacket használnak. Egy számot erre a veremre tenni a szám begépelésével lehet. A **FORTH** utasítások a bemenő paramétereket ezen a vermen várják, és ide teszik le az eredményeket. Ez azt jelenti, hogy a szükséges paraméterek az utasítás neve előtt már rajta kell legyenek a vermen (ezért is használ a **FORTH** RPN jelölést: 3+5 helyett 3 5 +). Így tehát valamilyen **PRINT 1000** helyett 1000.-ot kell írni (a . utasítás írja ki a verem legfelső elemét). Épp ezért az új **LOGO**-utasítások is így kell, hogy kapják az adatokat; pl. **FORWARD 100** helyett 100 **FORWARD**-ot kell írni, és így tovább.

Ez a **FORTH** csak egész számokat ismer, de a teknőcgrafikához szükség van szinuszokra és koszinuszokra; ezt egy rövid gépi kódú program (lista mellékelve) végzi, amely a **KOD** nevű **FORTH** szóban van elbujtatva; tulajdonképpen annyit csinál, hogy a **HD** (**HEADING**) változóban tárolt érték (fokban) szinuszának és koszinuszának ezerszeresét (csak egészek vannak a **FORTH**-ban!) a **SIN** és **COS** változóba pakolja, az **ARG** pedig előjelhelyessé teszi ezeket az értékeket.

Nem **LOGO** nyelvet írtam **FORTH**-ban, hanem teknőcgrafikát. Ez azt jelenti, hogy csak a rajzolóutasítások újak, a **FORTH** vezérlőszervezetek a régiek (meg szerintem jobbak is). Ezért a **FORTH**-teknőcgrafika használatkor alig fog hasonlítani a **LOGO**-ra. (Teknőcöt nem is rajzolunk.)

Ezért is ilyen rövid a program. Az egész mindenestül elfért 1024 byte-ban (ez 1 db ún. screen, **FORTH**-egység).

Varga Szabolcs,

1192 Budapest, Hungária út 6/a.

Az új utasítások szintaxisa:

n FORWARD	– a teknőc n egységgel elmozdul előre pl. 30 FORWARD
n BACK	– a teknőc n egységgel elmozdul hátra pl.: 30 BACK
n RIGHT	– a teknőc n fokkal elfordul jobbra pl.: 45 RIGHT
n LEFT	– a teknőc n fokkal elfordul balra pl.: 45 LEFT
PENUP	– nem húz vonalat a teknőc, miközben mozog
PENDOWN	– újra húz
n SETH	– beállítja a teknőc fejének az irányát: 90 SETH – észak, 270 – dél, 0 – kelet, 180 – nyugat, 45 – északkelet stb.
n SETX	– beállítja a teknőc helyének x, ill. y koordinátáját
n SETY	
x y SETXY	– mindkét koordinátát beállítja FIGYELEM! A LOGO -ban a SETXY , SETX , SETY utasításokkal is lehet rajzolni. Itt nem! (később esetleg)
KOZEP	– a teknőc középre megy, feje északra áll

A többi új szó: **X** és **Y** – konstansok; **SIN**, **COS**, **HD**, **PEN** – változók; **KOD** – a gépi rutin; **ARG** – az előjelhelyes rutin.

Ha túl hosszúak lennének a szavak, viszonylag könnyű rajta segíteni:

: FD FORWARD;
: BK BACK;
: RT RIGHT;
: LT LEFT;
: PN PENUP;

SCR #1

```

0 ( FORTH-TURTLE-GRAPHICS BY SZABOLCS VARGA 1987 )
1
2 23677 CONSTANT X 23678 CONSTANT Y 90 VARIABLE HD 0 VARIABLE SIN
3 0 VARIABLE COS 1 VARIABLE PEN CREATE KOD SMUDGE
4 197 C, 237 C, 75 C, HD , 205 C, 11563 , 62 C, 90 C, 205 C, 11560
5 , 239 C, 5 C, 163 C, 4 C, 49 C, 31 C, 164 C, 49 C, 49 C, 4 C,
6 4 C, 4 C, 1 C, 32 C, 164 C, 49 C, 49 C, 4 C, 4 C, 4 C, 56 C,
7 205 C, 11682 , 237 C, 67 C, COS , 205 C, 11682 , 237 C, 67 C,
8 SIN , 193 C, 195 C, NEXT , : SETX X C! ; : SETY Y C! ;
9 : ARG KOD HD @ DUP 180 > IF SIN @ MINUS SIN ! THEN DUP 90 > SWAP
10 270 < AND IF COS @ MINUS COS ! THEN ; : SETH HD ! ARG ;
11 : FORWARD DUP COS @ M* 1000 M/ SWAP DROP SWAP SIN @ M* 1000 M/
12 SWAP DROP PEN @ IF DRAW ELSE Y @ + Y ! X @ + X ! THEN ;
13 : LEFT HD @ + 360 + 360 MOD HD ! ARG ; : BACK MINUS FORWARD ;
14 : RIGHT MINUS LEFT ; : PENUP 0 PEN ! ; : PENDOWN 1 PEN ! ;
15 : SETXY Y C! X C! ; : KOZEP 127 87 SETXY 90 SETH ;

```

: PD PENDOWN;
: SX SETX;
: SY SETY;
: XY SETXY;
: SH SETH;
: KP KOZEP;

És ezek után a rövidítések ugyanúgy használhatók lesznek, mint az eredeti szavak (azok mellett).

Néhány példa az új

lehetőségek használatára

: NEGYZET KOZEP 5 1 DO 50 FORWARD 90 RIGHT LOOP; pl. NEGYZET négyzet tetszőleges helyre;
: N1 SETXY (két adatot vár) 0 SETH 5 1 DO 50 FORWARD 90 RIGHT LOOP; pl. 100 100 NEGYZET ötágú csillag akárhova;
: OT SETXY 0 SETH 6 1 30 DO FORWARD 144 RIGHT LOOP; pl. 150 100 OT egy kör akárhova;
: KOR SETXY 0 SETH 61 1 DO 6 FORWARD 6 RIGHT LOOP; pl. 120 120 KOR és így tovább.

Figyelmeztetés!

A TEKNŐCÖT NEM LEHET A KÉPERNYŐN LÁTNI RAJZOLÁS KÖZBEN! TUDNI KELL, HOL VAN, HOL ALLI

(Azért meg lehet tudni: X @.Y @. HD?-re kírja rendre az X, Y koordinátákat és a fejállást, de ekkor már az ember összefirkálta a képernyőt; ha meg letörli, úgyis mindegy, nem? – akkor a 0; 0-ba kerül.)

A szerkesztő azért van,

hogy a lap olyan legyen,

amilyenek az olvasói!

A gépi kódú rutin:

Hexa kód	Memória	Magyarázat			
08	8080	80	:80 elmentése		
08	18	HD	:HD elmentése		
08	08	CALL	ST-ROL-BC	:tetszőleges tetszőleges	
08	08	LD	0.90	:90 elmentése	
08	08	CALL	ST-ROL-BC	:tetszőleges tetszőleges	
08	08	ROT	40	:40 elmentése	
08	08	DB	8	:/	
A3	08	DB	163	:ST-ROL-BC	
04	08	DB	4	+	
11	08	DB	49	:DUP	
1F	08	DB	31	:SIN	
A4	08	DB	164	:ST-10	
31	08	DB	49	:DUP	
31	08	DB	49	:DUP	
04	08	DB	4	*	
04	08	DB	4	*	
04	08	DB	4	*	
01	08	DB	1	:EXCHANGE	
20	08	DB	32	:COS	
A4	08	DB	164	:ST-12	
31	08	DB	49	:DUP	
31	08	DB	49	:DUP	
04	08	DB	4	*	
04	08	DB	4	*	
04	08	DB	4	*	
38	08	DB	56	:END	
0D	A2	2D	CALL	FP-TC-BC	:Verem teteje
ED	43	COS	LD	(COS),BC	:--> COS
0D	A2	2D	CALL	FP-TC-BC	:Verem teteje
ED	43	SIN	LD	(SIN),BC	:--> SIN
01		PDP	BC		:BC visszamentése
08	04	3D	JP	NEXT	:Visszaugrás
					:FORTH-ú



KOMBINATÓRIKA, SZÁMOLÁS

1. feladat: Van 3-3 golyónk három színből.

Hányféleképpen válogathatunk ki közülük három darabot, ha egy-egy szint több példány is képviselhet? (Ismétléses variációk.)

```
10 N=3: SCLCLP
100 FOR I=1 TO N
110 FOR J=1 TO N
120 FOR K=1 TO N
140 PRINT CHR$(64+I);CHR$(64+J);CHR$(64+K);" "
150 WH=WH+1: IF K=N THEN PRINT;" "
160 IF J=N THEN PRINT
170 NEXT K,J,I
200 PRINT:PRINT WH;"DARAB VARIÁCIÓ"
```

F1. Háromnál többféle színből válasszunk három golyót!
F2. INPUT-tal adjuk meg a golyók színeit, és ezek kezdőbetűit frassuk ki a 140-es sorban.

2. feladat: Az előző feladat variációi közül válogassuk ki a permutációkat

(minden szín csak egyszer fordulhat elő)!

Az előző programban mindössze a

```
10 N=4: SCLCLP
100 FOR I=1 TO N
110 FOR J=1 TO N: IF J=I THEN 180
120 FOR K=1 TO N: IF I=K OR J=K THEN 170
140 PRINT CHR$(64+I);CHR$(64+J);CHR$(64+K);" "
150 PE=PE+1: IF K=N THEN PRINT;" "
160 IF J=N THEN PRINT
170 NEXT K
180 NEXT J
190 NEXT I
200 PRINT:PRINT PE;"DARAB PERMUTÁCIÓ"
```

módosításokat kell végrehajtani.

F3. Próbáljuk az egyenlőség-vizsgálatot „gyorsítani”. (Hat golyó esetén a fenti módszer bizony elég nehézkes.)

F4. Az előző feladat lehetőségeiből válasszunk ki azokat (jelöljük meg *-gal), amelyek ebben a feladatban is elfogadhatóak. Hasonlítsuk össze a variációk és permutációk számát!

F5. Engedjünk meg bizonyos ismétlődéseket! Például egy vagy két színből szerepelhessen legfeljebb két példány.

3. feladat: Öt különböző színű golyó közül válasszunk ki hármat, ahányféleképpen csak lehet! (Kombinációk.)

Ismét csak néhány sort kell változtatni:

```
10 N=5: SCLCLP
100 FOR I=1 TO N-2
110 FOR J=I+1 TO N-1
120 FOR K=J+1 TO N
140 PRINT CHR$(64+I);CHR$(64+J);CHR$(64+K);" "
150 FOR H=1 IF H=N THEN PRINT;" "
160 IF H=N THEN PRINT
170 NEXT K
180 NEXT J
190 NEXT I
200 PRINT:PRINT KO;"DARAB KOMBINÁCIÓ"
```

F6. Próbáljuk más paraméterekkel (színek, kiválasztott elemek száma) is futtatni a programot!

F8. Próbáljuk a F1. variációk közül megjelölni a kombinációkat!

F8. Engedjünk meg bizonyos ismétlődéseket!

4. feladat: Számítsuk ki N! értékét!

```
10 INPUT N
20 F=1
100 FOR I=2 TO N: F=F*I: NEXT I
120 PRINT F
```

Sajnos a fenti program csak 10!–15!-ig pontos. DBL típusú azonosítót használva (ha van ilyen: 20 DEFDBL F) N = 15–25-ig kielégítő. A Technika folyóirat 1984. VI. száma nagy pontosságú műveletekkel foglalkozik, többek közt N!-t tetszőleges pontossággal előállítja STRING-ek segítségével. Mutatunk erre egy másik megoldást is. Lehet, hogy ez bonyolultabb, de „rímél” a polinomok szorzása című (elég rossz határfokkal oktató) anyagrésze: A számjegyeket hármassával csoportosítjuk – A (J) – és a szorzást a csoportokkal egyenként végezzük. Az esetleges átvitel – A1 – a következő, J+1-ik csoporthoz adódik. A 180–188 sorok között a tagolt, jól olvasható kiírás történik (pontosan ezért csoportosítottunk hármassával). 195-ös sor a hatványozást, és a faktoriális választja szét.

```
50 INPUT "MINEK, HANYADIK HATV. (0,N)=-N FA  
KT.": A0,N
60 M=1: A(1)=1: A9=A0: IF A9=0 THEN A9=1
100 FOR I=1 TO N: FOR J=1 TO M: A(J)=A(J)*A9  
+A1
120 IF A(J)<1000 THEN A1=0: ELSE A1=INT(A(J)  
/1000): A(J)=A(J)-A1*1000
130 NEXT J
140 IF A1>.5 THEN M=M+1: A(M)=A1: A1=0
150 W=0: FOR J=M TO 1 STEP -1: A=A(J)  
182 FOR K=1 TO 3: E1=10*(3-K): B=INT(A/E1+.0  
1): A=A-E1*B: IF B+WC.001 THEN 185
183 W=1: CHAR 1,39-J*4+K,24,CHR$(48+B)
185 NEXT K,J
188 PRINT
195 IF A0=0 THEN A9=A9+1
200 NEXT I
```

F9. Könnyen átalakítható a feladat két tetszőlegesen nagy szám összeszorzására. Tulajdonképpen ez demonstrál egy igazi polinom szorzást (többtag+többtag).

F10. Próbáljuk meg a tizedespontot is kezelni!

F11. Próbáljunk meg osztani is!

5. feladat: Számoljunk ki egy tetszőleges binomiális együtthatót!

$$\binom{N}{K} = \frac{N!}{K!(N-K)!}$$
 segítségével a feladat visszavezethető az előzőre. Javasolunk egy másik megoldást is:

```
10 INPUT "N E'S K LEGYEN": N,K
12 K=INT(K): N=INT(N)
15 IF K=0 OR K=N THEN 10
20 F1=1: IF K=1 THEN F1=N-K
100 FOR I=1 TO K: F1=F1*(N-I+1)/I
120 NEXT I
170 PRINT F1
```

A 20-as sorban a Pascal háromszög szimmetrikus voltát használjuk ki:

$\left(\frac{9}{2}\right)$ gyorsabb, mint $\left(\frac{9}{7}\right)$!!

F12. Hasonlítsuk össze az együtthatók kétféle kiszámítását!

P O S T A



Hogyan lehet a C+14 USERPORT-ját 8 bites párhuzamos digitális bemenetként használni?

(A C 64-re vonatkozóan ráleltem a megoldásra: a CIA#2 B kapu adatirány-regiszteren DD03; 0 bit = IN a csatl. C-L lábain levő jelek adják DD01 bitjeit)

Timár Ferenc tanár, Pollák A. Erős-áramú SZKI, Szentés

A Plus/4-es a C 16-tól eltérően, és a C 64-hez hasonlóan rendelkezik egy úgynevezett felhasználói (előkelőbben: user) porttal. Ez mindkét gépnél a nyomtatott áramköri alaplmezéből a hátoldalon kialakított 2x12 érintkező sáv, melyek egymástól 3,96 mm távolságra vannak. Itt érhetők el a soros- és a párhuzamos átvitelt biztosító csatlakozási pontok.

De mint minden másban természetesen, a csatlakozó kialakításában is jelentős különbségek vannak a két gép között, és már az is csoda, hogy a csatlakozójuk azonos és legalább a tápfeszültségek és a soros átvitel majdnem valamennyi pontja azonos pozíción került kivezetésre. (Igy a C 64-hez gyártott szintillesztők – sajnos csak a külső műanyag dobozuk nélkül – de csatlakoztathatók a Plus/4-hez és működnek is vele). A soros vonalat végre egy ténylegesen erre való cél integrált áramkör kezeli, az amelyiket már a VC 20-ba is betervezték, de költségcsökkentés miatt még az oly annyira tisztelt C 64-ből is kispórolták, és egy szoftver-hardver öszvérrel helyettesítették.

Az új gépen tehát sokkal nagyobb átviteli sebesség érhető el. De a cég, hűen önmagához, ismét igyekezett megtakarítani amit csak lehetett, és a Plus/4-et egy szerényebb párhuzamos csatlakozási lehetőséggel látta el.

A C 64 a nyolc adat biten felül rendelkezik néhány további bittel is, amelyek segíthetnek az adatbitek kezelésében, illetve felhasználásában. Ez az úgynevezett közfogásos (handshake) üzemmódot is lehetővé teszi. Ilyen például a nagyobb kapacitású lemezmeghajtók vagy centronics illesztőjő nyomtatók esetén szükséges. Természetesen bármely más saját tervezésű áramkör csatlakoztatását is egyszerűbbé teszik (pl. eeprom programozó, analóg-digitál és digitál-analóg átalakítók stb.)

A Plus/4-nek csupán csak a nyolc adatbitje van meg, melyek vegyesen lehetnek be- illetve kimenetek. Ezt a C 64-nél már megkedvelt lehetőséget itt egy Commodore gyártmányú 6529 típusszámú integrált áramkör biztosítja olcsó, de szellemes módon.

Az áramkör tulajdonképpen egy nyolcbites tároló, amely a SFD10-FD1F cí-

mek bármelyikére aktivizálható. Az erre a címre írt értékek azonnal megjelennek az IC kimenetén és egy újabb érték beírásáig ott is maradnak. Ez tehát a „minden bit kimenet” üzemmód. A kiírt érték természetesen bármikor visszaolvasható.

A kimenetek kialakítása azonban olyan, hogy a logikai egy szinten levő bit kb. 0,7 mA árammal (2 LS terhelés) a logikai nulla szintre hozható. Mivel a visszaolvasás az IC tényleges kimenetéről, azaz lábairól történik, lehetséges ezen kivezetéseket bemenetnek használni.

Ez az oka annak, hogy a gép bekapcsoláskor minden bitet egybe állít, lehetővé téve, hogy bármelyik bemenet legyen.

Ha tehát csak bemenő bitekre van szükségünk nincs semmi más tennivalónk, mint a bitek olvasása. Ha csupa kimenő bite van szükségünk, akkor meg csak az értékeket kell beírunk a megadott címek valamelyikére.

Kicsit körültekintőbbnek kell lennünk a be- és kimenő bitek vegyes alkalmazásánál. A kiírandó értékeket úgy kell előállítani, hogy a kimeneti bitek értéke a kívánt érték, míg a bemeneti bitek értéke mindig logikai egy legyen. Jól használhatók erre a logikai utasítások mind basic-ből mind assembler-ből. A beolvasott értékből a kimeneti biteket figyelmen kívül hagyva pedig megkapjuk a bemeneti bitek értékét. Végül még arra hívom fel a magatofont is használók figyelmét, hogy D2-es biten a gép a magnetofon billentyűinek állapotát figyeli, és ezek lenyomása esetén ezt a bitet leföldeli azaz logikai nulla szintre kényszeríti.

Rendszeres olvasója vagyok a BIT-LET mellékletnek. Mint észrevettem, gyakran szerepel benne az ATARI 800 XL típusú gép is. Mivel én ilyen géppel rendelkezem azt szeretném kérdezni önöktől, tudják-e, hogy hol lehet ehhez a géphez könyveket kapni?

Szilágyi János, 2451 Ercsi Pf. 18/K
Úgy tűnik, a számítástechnikai könyvekkel is foglalkozó kiadók még nem ébredtek rá arra, hogy az Atari-gépek is egyre szélesebb körben terjednek Magyarországon. Az egyetlen, ezzel kapcsolatos kötet a Novotrade „Hetedhét Atari 800 XL” című kiadványa, Vitray Péter munkája – mást egyelőre ne is keressen a könyvesboltokban.

Olvasónk **Schuman Ádám**, (1121 Bp., Csorna u. 1.) írja:

Nekem C 16-os gépem van, amit igen jó gépek tartok. Mégis lépten-nyomon azzal találkozom, hogy a C 64 ennyivel meg annnyival jobb, a C 16 olyan rossz konstrukció, hogy már nem is gyártják.

Olvasónk 3 kérdése, és a rövid válaszok:

1. Tényleg abbahagyták a C 16 és a Plus/4 gyártását? Ha igen, miért?

A C 16 gyártását valóban leállították. Oka az, hogy ott van a nagyobb memóriájú, több beépített programot tartalmazó Plus/4, amely kompatibilis vele. A C 16 egyébként egyértelmű kudarc volt, a Plus/4 is majdnem. Fő oka: a szoftver hiánya. A mai számítógépiaccon gyakorlatilag az dönt, hogy milyen hamar, milyen minőségű és mennyiségű szoftver készül. Természetesen óriási előny, ha korábban meglevő programok változtatás nélkül futnak. (ld. IBM)

2. C 64-en már többször találkoztam beszélő, sőt éneklő programmal.

Lehet-e a C 16-ra is beszédszimulátor programot írni? (Ha esetleg van ilyen a Szerkesztőség birtokában, tegyék közzé!)

Elvileg nincs akadálya a beszédszintézisnek, bár nekünk nincs ilyen programunk. Miután a Spectrum (egyébként meglehetősen kevés szolgáltatást nyújtó) „hanggenerátorára” írt beszédszintetizátort is hallottunk már (sőt, meg is lehetett érteni), ezek után az ennél hardveresen is többet tudó C 16-on ennek nyilván nincs akadálya.

3. Hiányosság, hiba-e, hogy a C 16-on nincs SPRITE? Ha igen, mi helyettesíti?

Nem hiba. Az, hogy hiányosság-e, fel fogás kérdése, nyilván sokszor (különösen játékok esetében) lényegesen egyszerűbb az élet a SPRITE mellett. Szoftverúton azonban (mint sok minden egyebet, ezt is) lehet „szimulálni”, azaz lényegében szoftverrel helyettesíthető.

Sajnos, jelenleg ez sem áll rendelkezésünkre.

A számítógépprogram

nem kívánságaid,

hanem utasításaid

szerint működik!



Mielőtt eldönti, hogy kitölti-e, kérjük olvassa el a 17. oldali vezércikkünket.

A SZERKESZTŐ AZÉRT VAN, HOGY A LAP OLYAN LEGYEN, AMILYENEK AZ OLVASÓI
Éppen ezért kérjük, hogy szíveskedjen az alábbi kérdésekre válaszolni, s válaszait eljuttatni szerkesztőségünkbe.)

Szerintem fölöslegese a lapban a következő rovatok: (megfelelő aláhúzendő)

programajánlat – vallató – híroldal – posta – gépnyerő – hardverötletek – szoftverötletek – egyéb:

Jó lenne, ha több programot közölnének a lapban:

Spectrumra – Atarira – C 64-re – C 16-ra – Primóra – TVC-re – HT-re – egyéb:

Szívesen olvasna-e többet a lapban a magasabb kategóriájú gépekről, a számítástechnika csúcstechnológiájáról?

igen – nem

Vannak-e olyan rovatok, amelyeket szeretne gyakrabban vagy nagyobb terjedelemben látni a lapban? Ha igen, melyek ezek?

Egyetért-e azzal, hogy a BIT-LET ritkán közöl játékprogramokat, s akkor is csak logikai típusúakat?

igen – nem

Örülne-e ha egy olyan új rovatot indítanánk, amelyben közösen forgó játékprogramok megoldásait, térképeit, örökléteit adnánk közre.

igen – nem

Szívesen olvasna-e újabb Vallatókat, s ha igen, milyen gépekről?

Örülne egy olyan rovat létrehozásának, amelyben szoftvereket tesztelnénk? Ha igen, milyen típusú programokat kellene ebben a rovatban ön szerint vizsgálni? (Több is aláhúzható)

játékprogramok – felhasználói szoftverek – oktató programok – egyebek:

Jó lenne-e ha a BIT-LET rendszeresen közölné olyan cikkeket, amelyek az IBM és IBM kompatibilis gépekkel foglalkozóknak szólnának?

igen – nem

Kérjük, hogy amennyiben egyéb közölnivalója van, szíveskedjen ezt külön levélben csatolni a kitöltött ívhez.

A lapot kitöltő életkora:

foglalkozása:

EPROM SPECTRUMHOZ

A Spectrum-tulajdonosok nagy problémája, hogy a gyakran használt saját (nyomtató vezérlő, hibakezelő stb.) rutinokat állandóan be kell tölteni a memóriába. Ennek az elkerülése érdekében célszerű a programokat EPROM-ba égetni.

Ezt az EPROM-ot azonban a 65k-s memóriatartományon belül kell elhelyezni, mivel a Z80-as processzor másképpen nem éri el. Általában ezt vagy a RAM memória egy része helyett, vagy – egy kapcsoló áramkörrel – a ROM-mal párhuzamosan építik be. Mind a két megoldás hátránya, hogy így az EPROM és alatta lévő memória nem használható egyszerre.

Kevesen tudják azonban, hogy a ROM-ba 1170 byte egybefüggő szabad terület van. Ennek a területnek a kihasználására két lehetőség is van.

Az első, hogy egy 16k-s EPROM-ba a teljes ROM tartalmát át kell égetni, a megfelelő terület kivételével és ide a megfelelő programot kell beégetni. Ennek a megoldásnak az előnye, hogy a ROM program kisebb hibái is kijavíthatóak így. Hátránya azonban, hogy ha a mi programunk helyett akarunk ismét újabbat beírni, akkor újra cserélni kell az egész 16k-t így egy-két 1k-s rutinunk is rengeteg pénzt emészt fel.

HARDVER ÖTLETEK



A másik módszer, hogy csak egy kisebb, 1 vagy 2 k-s EPROM-ot használunk fel, és ezt kapcsoljuk egy külső áramkörrel a megfelelő címek esetén a ROM helyére.

Mivel a ROM-ban a szabad terület 386E–3CFF címek, ezért a címzés egyszerűsítése érdekében célszerű a 3800–386D terület tartalmát változtatlanul beégetni mind az 1, mint 2k-s EPROM-ba.

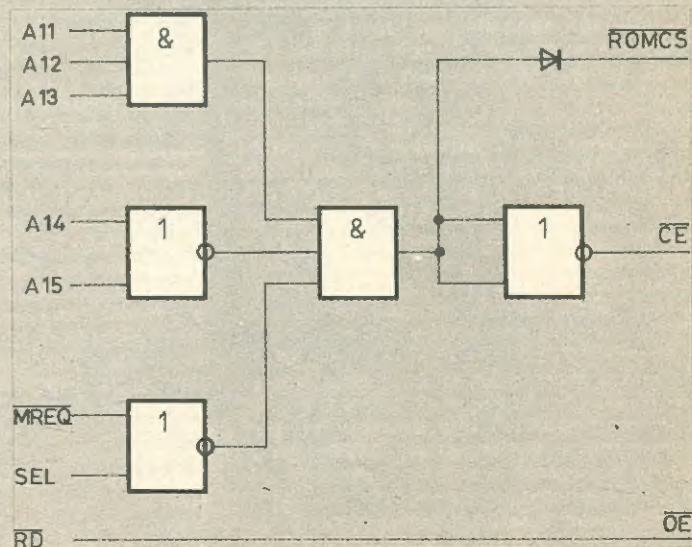
Így az 1k-s EPROM esetén (2758) 386E–3CFF közötti 914 byte-os terület használható fel szabadon.

A 2k-s EPROM előnye, hogy a 32–128 kódú karakterek formáját tetszőlegesen átdefiniálhatjuk szebb formájúra, a RAM foglalása nélkül. A karakterek formája a 3D00–3FFF területen van, így ezt a területet módosítással, vagy anélkül, de be kell égetnünk. Tehát ekkor a 386E–3CFF terület 1170 byte-ja használható fel.

A rajzon jelölt vezetékeken kívül természetesen be kell kötni a táp, föld, adat és címvezetéseket is.

A SEL jelű vezetékre 1k-s EPROM-nál az A10, 2k-s EPROM-nál az MREQ jelet kell kötni.

Krizsák László Hantos, Nagylóki u. 6.



Áts: OXFORD PASCAL C64-esen

– Novotrade, 221 o., 150 Ft.
(A több mint egy évtizede elterjedt, jól struktúrált, de a mikrogép-tulajdonosok által ritkán használt programozási nyelvről ad hasznos információkat a kötet.)

Kovács: Sakkprogramozásról mindenkinek – Novotrade, 241 o., 239 Ft.

(Valóban mindenkire szól a kötet: a kezdő sakkozótól, illetve a kezdő programozótól a legmagasabb szintig. Mindegyikük megtalálja azt az eljárást, algoritmust, ami felkészültségi szintjének megfelel.)

Szűcs Ervin: A számítógép tegnaptól holnapig – Műszaki Könyvkiadó, 136. o., 48 Ft.

Úgy tűnik, hogy a Műszaki Könyvkiadó „ráállt” a kezdőknek, a laikusoknak szóló számítástechnikai könyvek kiadására, amelyek nem a programozás mikéntjét próbálják mindenáron belesúlykolni az olvasók fejébe, hanem a számítógép mibenlétét, széles körű alkalmazási lehetőségeit igyekeznek ismertetni. Júniusban az „Első könyvem a mikrókról” című kiadványról szoltunk – amit a Novotrade-dal közösen jelentettek meg –, és máris itt az újabb, hasonló témájú könyv. Lehet, persze, hogy ez az egybeesés csak a nyomdai átfutás szeszélyének köszönhető, de mindenestre hasznos, hogy az olvasó igényei szerint válogathat a hasonló témájú könyvek közül.

Ami tovább erősíti a két kötet hasonlóságát: tulajdonképp mindkettő képeskönyv. Míg azonban az „Első könyvem...” a szövegbe beépített színes képekkel illusztrálja mondanivalóját – illetve a képek teszik ki a kötet nagy részét –, addig „A számítógép...” szerkesztési koncepciója más: minden második oldalon egy-egy egészlapos (fekete-fehér) illusztráció, a következőn pedig a hozzá tartozó szöveges fejezet.

Ezt a szerkesztési koncepciót követi az egész kötet: ha felüjtjük, a bal oldali lapon a képet látjuk, jobb oldalon pedig a hozzátartozó szöveg olvasható. A szerző a kötet elején így fogalmaz: „Talán sikerül a szemléltető képek és a hozzájuk tartozó szöveg segítségével együttesen meggyőzni az Olvasót arról, hogy a kultúra egy és oszthatatlan, és hogy annak ma már szerves része a technikai kultúra, s hozzá tartozik a számítástechnikai kultúra is.” Mindezt valamivel idősebb korosztállyal igyekszik tudatosítani, mint a korábban említett kötet – erre utal a nagyobb terjedelem, a komolyabb, elemző hangnem is. Csakhogy...

A kötet érdemi része, azaz a szöveg ellen nem lehet érdemi kifogásunk. A képek azonban... Nos, a kötet grafikai kidolgozása csapnivaló – és ez annál nagyobb hiba, mert a szerző is egyenlő fontosságot tulajdonított a képeknek és a szövegnek. A képek egy



részt rajzok, másik részét fényképek, harmadikat pedig e kettő kombinációja teszi ki. A karikatúraszerű rajzok jópofák, és érzékeltetik is a szemközi oldal tartalmát – nem is ezekkel van bajunk.

A fotók azonban annyira homályosak, szürkék, tónustalanok, hogy az már elfogadhatatlan. Ahol pedig a fotókat a rajzokkal kombinálják, ott – nincs rá jobb szó – borzalmas eredmények születtek. Filc- vagy tustollal ábrázolt sesznű fotók próbálnak illeszkedni a hozzájuk erőltetett grafikákhoz.

A háborgás után nézzük azonban az érdemi részt, a szöveg tartalmát! Azt már említettük, hogy a grafikus és a szöveges oldalak felváltva követik egymást – de azt még nem, hogy minden szövegoldalnak külön címe is van. Nem rossz ez, hiszen így minden fejezet egészoldalas illusztrációt kap – de vannak hátulütői is. A „fejezetnek” megfelelő egyoldalas szöveg néhol kissé terjengős – ez fordul elő, persze, ritkábban, hiszen az egyoldalas megkötés hatalmas korlátot jelent ilyen témakörök esetében –, máshol viszont a szerző igyekszik szinte az érthetlenségig tömöríteni mondanivalóját, vagy pedig több, egymást következő fejezetnek hasonló címet adva folytatja ugyanannak a területnek az ismertetését.

Ennek ellenére jó áttekintést ad a kötet a számítástechnika múltjáról, történetéről, fejlődéséről, és jelenlegi helyzetéről. Mondanivalója lényegének tekinti a szerző az algoritmus fogalmának, alapjainak megismertetését.

A kötet lényegében négy fő részre osztható. (Erre talán a szerző sem gondolt, könyvét 65 részre osztva. Így lehet, hogy felosztásunk esetleges, önkényes.) Az első, legvaskosabb szakasz a számítástechnika történetével foglalkozik. Itt a könyv a feltételezhetően tizenéves olvasók számára jó áttekintést nyújt a számítástechnika kialakulásáról, a XX. század elejéig. A – szerintünk – összefoglalást jelentő „A technika fejlődése” fejezetben olvasható vulgár-marxista ér-

telmezést viszont valószínűleg már a tizenévesek is idegenkedve olvassák. A második rész egyfajta elméleti tudnivalók gyűjteményének tekinthető: szó esik itt a számítógépek programozásának alapelveiről, a gépek által használt algoritmusok alkalmazásáról, a számítógép-generációkról, az információk feldolgozásáról. Itt is lehetnek persze kifogásaink a tartalommal szemben. Így az információval, informatikával foglalkozó fejezetek meglehetősen terjengősek, ugyanakkor nem sokat mondanak – hogy mi is információelméleti kifejezéssel éljünk: redundánsak, azaz sok ismétlődő ismeretet tartalmaznak.

A harmadik rész foglalkozik az alkalmazásokkal – és ez a könyv leghasznosabb része: az általánosan vett számítógépes termelésirányítástól kiindulva a szerző sorra veszi a felhasználási lehetőségeket egészen a sportig, művészetig, játékidőig. Eközben mindig hangsúlyozza: a gép csak eszköz az ember kezében, csak arra képes, amire programozzák. Jelentős terjedelmet szentel itt a szerző a robotok fejlesztésére, generációik ismertetésére – sőt a sci-fi szerzők ezzel kapcsolatos véleményére is.

Az utolsónak tekinthető néhány oldalas negyedik rész a számítógépek további fejlődésével foglalkozik. A szerző itt két képet villant fel: az egyiknél meglehetősen közhelyszerűen egy nem túl régi japán kiállítás tapasztalataira hivatkozik, (napelemek, alakelemzők, TV, lemezjátszó, háztartás automatizálása) mint az eljövendő nagy változásra. Vagyis a szerző ettől reméli az ipar megváltását? A másik kép, a robotok támadásával kapcsolatban – a „Rémképek” címet viseli – a Bibliától József Attiláig idéz írókat, költőket, arra a végkövetkeztetésre jutva, hogy a gépek az embert szolgálják.

Végül is a szerző még a legeslegutolsó, mindent összefoglaló fejezetben sem osztja el a kötet tizenéves olvasóinak rémálmait: nem győzi meg őket a robottámadás lehetetlenségéről – de ez talán nem is volt feladata.

Tallér József

AZ ATARI-NYERŐ 3. FELADATÁNAK MEGOLDÁSA

A közölt program két, maximum 60 jegyű számot szoroz össze. A szorzást a gyorsaság kedvéért 1000-es számrendszerben végzi. Először a bekérdezett számokat átváltja 1000-es számrendszerbe, majd először elvégzi a szorzást úgy, hogy a maradékot nem viszi át, itt tehát 1000-nél nagyobb (maximálisan 19960020) „számjegyek” keletkeznek. A következő lépésben történik jobbról balra a maradékok átvitele. Ezután már csak ki kell írni a szorzatot. Az 1000-es számrendszer az indokolja, hogy egyrészt ebbe könnyű átírni (és innen visszafütni) egy tízes számrendszerbeli számot, másrészt ez a legnagyobb az ilyen tulajdonságú alapszámok közül, mivel már 15 db maximum 9999*9999 nagyságú szám összeadásakor a gép kerekít. (Mi a programot C 16-ra írtuk, ha valakinek a gépe már 20 db 999*999 nagyságú számot sem tud pontosan összeadni, annak át kell dolgozni a programot 100-as számrendszerre.) A gyorsítást elsősorban azért ériük el, hogy a szorzat jegyeinek kiszámításakor max. 3600 helyett csak max. 400 szorzást végzünk el. (A gép két változóban tárolt 3 jegyű számot kb. ugyanannyi idő alatt szoroz össze, mint két egyjegyűt!)

Nézzük a programot soronként!

```
10 DIMT(2,20),E(41)
```

A 10-es sorban dimenzionáljuk a T tömböt, melynek két „sorában” a két

összeszorozandó számot tároljuk 1000-es számrendszerben, „hátról előre”, tehát az első helyen az utolsó számjegyet; s az E tömböt, mely hasonló módon fogja tartalmazni a szorzatot.

```
20 FORI=1TO2
30 INPUTA$:L=LEN(A$):L(I)=L+2)/3:IFL>60THEN30
40 FORJ=1TOL(I):T(I,J)=VAL(MID$(A$,L+3-3*J,3)):NEXTJ
50 PRINT:PRINT:PRINT:PRINT
```

A 20-50-es sor bekéri a két számot, s 1000-es számrendszerbe átalakítva eltárolja őket a T tömbben. Ha 60 jegynél hosszabb számot írunk be, újra kérdez. L (1) ill. L (2) az első, ill. második szám hossza lesz 1000-es számrendszerben (ill. az egész részük lesz a hossz, de csak ciklus felső határaként használjuk őket, így ez nem zavar).

A 40-es sorban végzzük az átalakítást, a „00” string A\$ elé írására csak az első számjegy meghatározása miatt van szükség.

```
60 FORH=1TOL(2):FORJ=1TOL(1):E(J,H-1)=E(J,H-1)+T(1,J)*T(2,H):NEXTJ,H
```

A 60-as sor számítja a szorzat „számjegyeit” (maradékvitel nélkül), itt jól jön, hogy a számokat „hátról előre” írtuk fel, különben az egyes számjegy-szorzatok helyét nehéz lenne meghatározni. Megjegyezzük, hogy itt (is) használjuk a BASIC ama tulajdonságát, hogy RUN hatására a változók nullázódnak; ha a programot szubrutinként szeretnénk használni, akkor ezekről a nullázásokról is gondoskodni kell!

```
70 FORJ=1TO40:T=INT(E(J)/1000):Z=E(J)-1000*T:E(J)=Z:E(J+1)=E(J+1)+T:NEXTJ
```

A 70-es sor teszi helyére a maradékokat, a szorzatban jobbról balra haladva.

```
80 K=-1:FORJ=41TO1STEP-1:IFE(J)=0ANDKTHENNEXTJ:PRINT0:END
90 PRINTRIGHT$( " "+STR$(E(J)-(K=0)*1000),3):K=0:NEXTJ:END
```

A szorzat kiíratását a 80-90 sorok végzik, a K kapcsoló mutatja, hogy elkezdjük-e már a kiíratást. Erre azért van szükség, mert a szám elején 0-kat nem szeretnénk kiíratni. A ciklus a 80-as sorban csak úgy tud befejeződni, ha a szorzat 0, ekkor ezt a 0-t ki kell írni!

A 90-es sor ír ki egy valódi 1000-es számrendszerbeli számjegyet, ha ez az első kiírandó ilyen számjegy, s nem 3-jegyű, akkor nem kell 0-kat írunk az elejére, a későbbi számjegyek esetén viszont kell. Ezt végzi el a K kapcsoló értékétől függően a RIGHT\$ függvény.

A program futását kis szorzandók esetére gyorsíthatnánk egy picit úgy, hogy a 70-es és a 80-as sorokban a 40, ill. 41 (amely helyett elég lenne 40-et írni, két, maximum 60-jegyű szorzata maximum 120 jegyű) helyett INT(L(1))+INT(L(2))-t írának.

GÉPNYERŐ! JÖNI GÉPNYERŐ! JÖNI GÉPNYERŐ!

A gépszerző tárgyalások úgy tűnik, eredménnyel járnak, így ígérhetjük, hogy jövő hónapban ismét indítunk egy három-négy hónapos pályázatot, amelynek díja egy gép lesz. (A helyzet az, hogy lapzártáig sem sikerült a kipontozott részt kitölteni, így hát marad a titok, titok egy újabb hónapig.) Addig is bemelegítésnek egy jó kis feladat, amely nem könnyű, de amelyhez kellően értékes díjat is ajánlunk:

lapaszfalat

N Y E R O

1. díj: 3000 forintos Centrum-utalvány
2. díj: 1000 forintos Centrum-utalvány
- 3-7. díj: 200-200 forintos Centrum-utalvány

A feladat a következő:

Úgy hisszük, hogy a bűvös kockát (az eredeti, 3*3*3-as kockát) mindenki ismeri, így leírásától eltekintünk. A feladat ezzel kapcsolatos lesz, nem egy teljes bűvös kocka programot kell írni, csak annak egy részét.

A feladat két részből áll:

Találjanak ki egy adatstruktúrát, melynek segítségével a számítógépben tárolhatják a bűvös kocka egy-egy helyzetét. Az adatstruktúrát szöveggel kérjük leírni, s röviden megindokolni, hogy miért ezt választották. A választásnál a fő szempont az, hogy az adatstruktúra minél jobban megfeleljen egy esetleges bűvös kocka programnak (mely a kockát tetszőleges elforgatott helyzetből alapállapotba „tekeri” vissza).

Írják meg (BASIC-ben) a program elejét, amely a megfelelő tömböket dimenzionálja, esetleg segédváltozókat vezet be, s ad nekik értéket; valamint egy szubrutint, mely inputonként a kocka egyik lapjának megfelelő belső kódot és egy 1 és 3 közé eső számot kap, s az adatstruktúrán „elvégezi” a megfelelő oldalnak az óramutató járásával egyirányú 90, 180, ill. 270 fokos elforgatását. **A programot papíron kérjük beküldeni!**

lapaszfalat

N Y E R O

Kérjük levágni és a levélre
felragasztani! Beküldési
határidő: 1987. augusztus 31.